Heuristic Reasoning on Graph and Game Complexity of Sudoku

Zhe Chen
LATTIS, INSA, University of Toulouse
135 Avenue de Rangueil, 31077 Toulouse, France
Email: zchen@insa-toulouse.fr

Abstract

The Sudoku puzzle has achieved worldwide popularity recently, and attracted great attention of the computational intelligence community. Sudoku is always considered as Satisfiability Problem or Constraint Satisfaction Problem. In this paper, we propose to focus on the essential graph structure underlying the Sudoku puzzle. First, we formalize Sudoku as a graph. Then a solving algorithm based on heuristic reasoning on the graph is proposed. The related r-Reduction theorem, inference theorem and their properties are proved, providing the formal basis for developments of Sudoku solving systems. In order to evaluate the difficulty levels of puzzles, a quantitative measurement of the complexity level of Sudoku puzzles based on the graph structure and information theory is proposed. Experimental results show that all the puzzles can be solved fast using the proposed heuristic reasoning, and that the proposed game complexity metrics can discriminate difficulty levels of puzzles perfectly.

1. Introduction

The Sudoku puzzle has become popular in Japan since 1986, and became world-wide popular in 2005. The completion rules of the puzzle are rather simple, however the required reasoning techniques for solving the puzzle are often complex. Because Sudoku puzzle is a good example of various reasoning approaches in Artificial Intelligence area, it has attracted great attention of the computational intelligence community [1][5][7][10][13][14].

Fig. 1 is an example of the Sudoku puzzle. The left part provides a Sudoku puzzle, and its solution is shown on the right. We define the notations as follows:

Definition 1: The **Sudoku puzzle** is a 9×9 **grid**, which comprises nine 3×3 **sub-grid**s, which comprises 3×3 **cell**s. Some cells are filled with numbers from 1 to 9, whereas others are left blank.

Definition 2: The Sudoku puzzle is **solved** by filling all the blank cells with numbers from 1 to 9, such that every row, every column and every sub-grid contains each of the nine possible numbers. Sudoku puzzle has exactly one solution.

There are mainly three directions in the Sudoku puzzle studies:

1. Sudoku generation. One question is how to generate randomly a Sudoku puzzle which has exactly one solution. Another question is the estimation of the total number of the possible Sudoku puzzles [2][3][11].

2. Sudoku solving. Up to now, most of the studies are on this direction. People proposed several practical reasoning algorithms based on different knowledge representations. [15] shows that the generalized Sudoku puzzle is NP-Complete. [5][7][14] encode Sudoku as a Satisfiability Problem (SAT), and use a general SAT solver to get the solution. [10][13] represents the Sudoku puzzle as a Constraint Satisfaction Problem (CSP), and compares different propagation schemes for solving Sudoku. However, [5] reported that some hard puzzles cannot be solved using these methods, such as [6][9]. In this paper, we will formalize Sudoku as a graph, and develop a solving algorithm based on heuristic reasoning on the graph.

3. Sudoku complexity evaluation. Most of the Sudoku designers classify the difficulty levels of Sudoku puzzles as, for instance, "easy", "intermediate" or "hard" (or levels 1, 2, 3) according to their experiences. However, up to now there are few publications on this subject. [7] only mentioned that the difficulty level depends on the number of initial non-blank cells provided. This is a very rough evaluation, and we will propose a more accurate quantitative metric for difficulty evaluation based on the complexity of the underlying graph structure.

This paper deals with the second and third aspects, in order to provide a formal basis for the studies and developments of Sudoku solving systems and difficulty evaluation. In section II, the graph structure underlying Sudoku puzzle is defined. In section III, heuristic reasoning on the graph is proposed, and its related theorems, properties and algorithm are explained in detail. Section IV focuses on how to evaluate the game complexity of the Sudoku puzzle based on information theory. To assess our theory, experiments are performed and the results are discussed in section V. Conclusion and future works are presented at the end.

2. The Sudoku Puzzle and Graph

The Sudoku puzzle is essentially a directed graph. The cells are vertices, and the constraints are directed edges. Fig. 2 shows part of the Sudoku graph related to $cell_{45}$ of the

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Figure 1. A Sudoku Puzzle

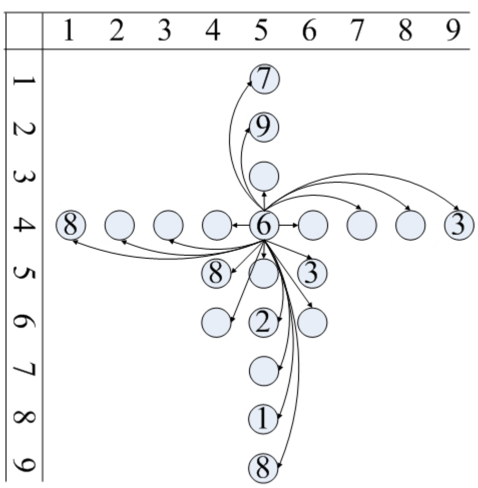

Figure 2. Part of the Sudoku Graph

puzzle in Fig. 1. The cells in the same row, the same column and the same sub-grid as $cell_{45}$ are vertices in the subgraph, and the edges represent the constrain relationships between $cell_{45}$ and the connected cells. Notice that the full graph is constructed by 9×9 such subgraphs that are connected to be the entire graph. Initially, $cell_{45}$ is filled with number 6, so we say that the set of possible values of the vertex is $v_{45} = \{6\}$, and the filled value is $c_{45} = 6$.

In Fig. 1, the vertex v_{13} may be filled with 1, 2, or 4, so the candidates set is $v_{13} = \{1, 2, 4\}$, and the size of the set is $|v_{13}| = 3$. In the reasoning process, if we find that $cell_{31}$ should be filled with 1, then we can remove 1 from the candidates set v_{13} (they are connected by an edge $e_{31,13}$), so $v_{13} = \{2, 4\}$. That is, the information of possible candidates in each cell is propagated through the edges on the graph, and change the values of candidates sets of connected vertices. The graph underlying the Sudoku puzzle is formally defined as follows.

Definition 3: The **Sudoku alphabet** is $\sum = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Definition 4: The **vertex candidates set** is $v_{ij} = \{x | x \in \sum$, and $cell_{ij}$ is possible to be filled with the number x with the information we have$\}$.

Definition 5: If $|v_{ij}| = 1$, the vertex $cell_{ij}$ is **filled** with the number $c_{ij} \in v_{ij}$.

Definition 6: The **Sudoku graph** is $G = (V, E)$, where the vertices set $V = \{cell_{ij} | cell_{ij}$ is the cell in i-th row and j-th column, $1 \leq i, j \leq 9\}$, the set of edges $E = \{(cell_{ij}, cell_{kl}), (cell_{kl}, cell_{ij}) |$ there is the constraint $c_{ij} \neq c_{kl}$ between $cell_{ij}$ and $cell_{kl}\}$. For simplification, edge $(cell_{ij}, cell_{kl})$ is noted as $e_{ij,kl}$.

Definition 7: **Initiation** of Sudoku Graph: $\forall cell_{ij}$, if $cell_{ij}$ is filled with number k in the puzzle as initial, then $c_{ij} = k$, $v_{ij} = \{k\}$.

Both the solving algorithm and game complexity proposed in this paper are based on the graph structure. Ideally, player can solve the puzzle in this way: player starts from any one of the vertices, traverses all the vertices once and only once, along with the directed edges. Each time when the player arrives at a vertex $cell_{ij}$, he or she want to choose the correct number $x \in v_{ij}$ to fill the cell $c_{ij} = x$, and output the number. The player continues to traverse the remaining vertices by choosing one of the edges, until all the vertices are visited. At the end, the output of numbers is the solution.

However, how to "choose the correct number $x \in v_{ij}$" is the essential uncertain problem. In the next section, a heuristic reasoning approach is proposed to achieve this goal. And the difficulty level of the Sudoku puzzle is evaluated based on the concept of "uncertainty of choice" at each vertex of the graph, since it is related to the magnitude of uncertainty in the reasoning.

3. Heuristic Reasoning on the Graph

In order to present the heuristic reasoning on the Sudoku graph, we define the concept of "section" of a Sudoku graph.

Definition 8: **Section** S is a complete subgraph in the Sudoku graph G.

"Complete subgraph" is a basic concept from discrete mathematics. The definition means that a section is a subgraph of graph G, such that every two vertices in a section are directly connected by two directed edges of opposite directions. The vertices in a row, a column or a sub-grid construct totally 27 "sections". Moreover, they are the only sections in a Sudoku graph. Each vertex belongs to 3 sections.

For heuristic reasoning, the most obvious fact is that if a cell is filled with a number, then the cells in the same section cannot be filled with the same number. So this number can be removed from the candidates sets. Because we only remove one element from the sets, the process is called 1-Reduction.

Theorem 9 (1-Reduction): $\forall i, j, k, l$, if $\exists |v_{ij}| = 1$, and there is an edge $e_{ij,kl}$, then $c_{kl} \in v_{kl} - v_{ij}$.

proof: According to Definition 5,

$$c_{kl} \in v_{kl} \tag{1}$$

There is an edge $e_{ij,kl}$, according to Definition 6, $c_{ij} \neq c_{kl}$. So,

$$c_{kl} \in v_{kl} - \{c_{ij}\} \tag{2}$$

$|v_{ij}| = 1$, so $v_{ij} = \{c_{ij}\}$, we conclude $c_{kl} \in v_{kl} - v_{ij}$. □

Using this theorem, once a cell is filled with number c_{ij}, then we can remove the number from the candidates sets of the connected vertices.

However, sometimes we are not able to find such a vertex having $|v_{ij}| = 1$, but we are able to find r vertices that share r possible numbers. In this case, we can remove the r numbers from the candidates sets of the other vertices that are in the same section as the r vertices. Notice that r numbers are reduced from the sets. To formalize the information propagation on the graph, we have a more general reasoning theorem:

Theorem 10 (r-Reduction): In any section S_k, if $\exists cell_1$, $cell_2$, ..., $cell_r \in S_k$ such that $|v_1 \cup v_2 \cup ... \cup v_r| = r$, then $\forall cell_{ij} \in S_k$ and $cell_{ij} \notin \{cell_1, cell_2, ..., cell_r\}$, we have $c_{ij} \in v_{ij} - v_1 \cup v_2 \cup ... \cup v_r$.

proof: Vertices $cell_1, cell_2, ..., cell_r, cell_{ij}$ are in the same section, so $c_1, c_2, ..., c_r$ and c_{ij} are all different,

$$|\{c_1, c_2, ..., c_r\}| = r \tag{3}$$

$$c_{ij} \in v_{ij} - \{c_1, c_2, ..., c_r\} \tag{4}$$

According to Definition 5,

$$\{c_1, c_2, ..., c_r\} \subseteq v_1 \cup v_2 \cup ... \cup v_r \tag{5}$$

So,

$$|\{c_1, c_2, ..., c_r\}| \leq |v_1 \cup v_2 \cup ... \cup v_r| \tag{6}$$

with equation (3) and assumption in the theorem, we have,

$$r = |\{c_1, c_2, ..., c_r\}| \leq |v_1 \cup v_2 \cup ... \cup v_r| = r \tag{7}$$

it implies that in equation (5),

$$\{c_1, c_2, ..., c_r\} = v_1 \cup v_2 \cup ... \cup v_r \tag{8}$$

substitute for equation (4), we conclude:

$$c_{ij} \in v_{ij} - v_1 \cup v_2 \cup ... \cup v_r \tag{9}$$

□

This theorem allows player to remove a set of numbers from the candidates set of a vertex, depending on the candidates sets of connected vertices in the same section.

Theorem 10 is a very strong reduction rule for the Sudoku puzzle reasoning. In experimental results, we will show that it can solve nearly all the Sudoku puzzles, except some very hard puzzles. Because these very hard puzzles provide less redundant pieces of information for heuristic reasoning [5]. In order to solve all the puzzles, we have to add the following Inference Rule. This theorem allows player to make additional assumptions, which can increase the redundant information for reasoning.

Theorem 11 (Inference): $\forall v_{ij} = \{c_1, c_2, ...\}$, if we set $v_{ij} = \{c_k\} \subseteq \{c_1, c_2, ...\}$, then if r-Reduction can reduce the puzzle to a solution, then we conclude $c_{ij} = c_k$.

proof: As we know, $c_{ij} \in v_{ij}$ is a single number. The puzzle has exactly one solution, so c_{ij} is in the solution. Because we have set $v_{ij} = \{c_k\}$, $cell_{ij}$ is surely filled with c_k in the solution. So we conclude $c_{ij} = c_k$. □

Based on Theorem 10 and Theorem 11, we propose the following algorithm that can solve all the Sudoku puzzles in nearly polynomial time.

Algorithm 12 (Algorithm for solving Sudoku):

Initiation: set global variables $v_{11}, ..., v_{99}$, according to the filled cells in Sudoku.

```
bool Solver()
{
   while(1)
   {
      Reduction();
      if (Sudoku is not solved)
         Inference();
      else if (Sudoku has no solution)
         return false;
      else return true;
   }
}

Reduction()
{
   r=1;
   while(r < 9){
      r-Reduction;
      if(no candidates set is reduced)
         r++;
```

```
        else r=1;
        }
}

Inference()
{
    Select randomly a vertex cell_ij which was not selected
        and has |v_ij| > 1;
    for each x ∈ v_ij
    {
        set c_ij = x;
        if( Solver() ) return;
    }
}
```

As we mentioned in Theorem 10, the "r-Reduction" statement in function "Reduction()" tries to find r vertices that share r numbers for reduction, for each section in the 27 sections. Its computational complexity is polynomial.

Notice that if we only use the inference rule, the reasoning process actually generates a backtracking inference tree, with exponential complexity. However, if r-Reduction can solve puzzles with few inferences (e.g., 1 or 2 times), the practical computational complexity is still nearly polynomial. As we can learn from the algorithm, we only use the inference as an assistant of r-Reduction. As we will show in the experiments, the practical computational complexity is almost equal to the polynomial complexity of r-Reduction.

4. On the Game Complexity of Sudoku

The game complexity of Sudoku depends on the initial distribution of the numbers in the cells [7]. This section aims at providing a quantitative metric to assess the complexity without any playing. In [12], information entropy is proposed to evaluate the uncertainty of signals sent by an information source. The "uncertainty" concept is similar to the uncertainty of choosing the correct number for the vertex when the player is traversing the Sudoku graph, by considering the possible choices on a vertex as the possible signals to be sent in a state of information source. So we propose to evaluate the complexity of solving Sudoku puzzles using the information entropy.

On the player's viewpoint, there are $|v_{ij}|$ possible choices for each vertex $cell_{ij}$. At the first step, the player has to choose one from the $|v_{ij}|$ possible choices. Then, at the second step, the player chooses to go to one of the 20 connected vertices in the same row, column and sub-grid (see Fig. 2). Taking the two steps together, the player has $20 \times |v_{ij}|$ choices ($(c_k, e_{ij,lm})$ pairs). The choices are of the same probability, because the player can only try them one by one without additional hints. So for each vertex v_{ij}, the entropy is $H_{ij} = -\log \frac{1}{20|v_{ij}|}$.

For the entire graph, the entropy is the average entropy of all the 81 vertices:

$H = -\frac{1}{81}\sum_{ij} \log \frac{1}{20|v_{ij}|} = -\log \frac{1}{20} - \frac{1}{81}\sum_{ij} \log \frac{1}{|v_{ij}|}$

The first part "$-\log \frac{1}{20}$" of the formula represents the intrinsic complexity of the puzzle which is associated with the structure (20 connected vertices). Then the later part of the formula expresses the complexity related to the distribution of the initial values. For simplification, we remove the constant part $-\log \frac{1}{20}$ from the equation, and define the Sudoku entropy and the game complexity as follows.

Definition 13: The **Sudoku entropy** is defined as

$$H_G = -\frac{1}{81}\sum_{ij} \log \frac{1}{|v_{ij}|} \tag{10}$$

Definition 14: The **game complexity** is the Sudoku Entropy after initiation plus one 1-reduction.

The difficulty for players to solve the puzzle is based on the Sudoku entropy. The game complexity for players is the Sudoku entropy after the initiation of the graph, and removing the numbers from vertex candidates set which are obviously impossible choices.

For example, in Fig. 1, after initiation and one 1-reduction, $cell_{31}$ has the candidates set $\{1, 2\}$, so $H_{31} = -\log \frac{1}{2}$. Another vertex $cell_{13}$ has the candidates set $\{1, 2, 4\}$, so $H_{13} = -\log \frac{1}{3}$.

5. Experimental Results

In this section, we show the effectiveness of our algorithm for solving Sudoku and of the quantitative metrics for Sudoku complexity estimation.

Up to now, since difficulty levels are defined by experts, there is no tool based on an analytical approach for Sudoku estimation. So we have created test sets of three difficulty levels. The "easy" level and the "intermediate" level include both 24 instances from the first 3 books of easy and intermediate Sudoku puzzles respectively in [4].

According to [7], there are no known puzzles with 16 pre-assigned cells, because they lead to more than one solution. So the puzzles with only 17 pre-assigned cells are the most difficult puzzles. When additional cells are assigned, the information redundancy increases, and the complexity decreases. [8] contains 47,621 Sudokus of exact 17 pre-assigned cells. Our "hard" level data set includes 10,000 puzzles from them.

Using the reasoning algorithm and the game complexity metric based on entropy, the results of the 3 levels are compared in Table 1. The r-Reduction rule is strong enough to solve all the "easy" and "intermediate" puzzles, and 70.5% of the "hard" puzzles.

The values of game complexity discriminate the actual difficulty levels of puzzles perfectly. The easy puzzles have game complexity values in the domain [0.5362, 0.9046], which are all less than the values of intermediate puzzles

in the domain [1.2246, 1.3965], which are all less than the values of hard puzzles in the domain [1.6946, 1.8189]. Therefore, our quantitative estimation is highly in accordant with the evaluation of the experts who design the puzzles.

	Easy	Intermediate	Hard
Solved (r-Reduction only)	100%	100%	70.5%
Solved (r-Reduction + Inference)	(100%)	(100%)	100%
Minimum Game Complexity	0.5362	1.2246	1.6946
Maximum Game Complexity	0.9046	1.3965	1.8189
Average Game Complexity	0.7439	1.3049	1.7526

Table 1. Results on 3 levels

The detailed results of "hard" puzzles are reported in Table 2. If we only use r-Reduction, 70.5 percent of the puzzles can be solved. If the inference rule is allowed, all the puzzles can be solved with only average 1.6 inferences for a puzzle. So, as explained in section 3, this keeps the practical computational complexity of the algorithm in nearly polynomial time. There is nearly no difference on time consumed between the situations when the inference rule is enabled or not.

The average game complexity of the unsolved puzzles is greater than that of the solved puzzles, which proves again that our metric can discriminate the difficulty levels of Sudoku puzzles.

People are trying to produce more diabolic puzzles to challenge the reasoning ability of computers. For example, [5] reported that many solvers and algorithms fail to solve the problems from [6] [9], such as "AI Escargot" puzzle. Using our algorithm, the computer can solve "AI Escargot" with only 91 inferences, which is claimed to be "the most difficult Sudoku puzzle known so far". Moreover, the 21 puzzles from [9] are solved with 2442 inferences, average 116 inferences for one puzzle. These results prove that our algorithm can solve all the known puzzles at any difficulty level with high efficiency.

6. Conclusion and Future Works

In this paper, we provided a formal basis for studies and developments of Sudoku solving systems and complexity evaluation. We defined the formal graph-based knowledge representation of the Sudoku puzzle, and proposed a heuristic reasoning algorithm based on the theorems of r-Reduction and Inference. The r-Reduction theorem is a very strong heuristic reasoning rule for Sudoku puzzles, and we have showed that it can solve most of the puzzles itself. In order to solve the puzzles more complex, we proposed to add the inference theorem as assistant in the algorithm. Because the r-Reduction rule is strong enough, few inferences are needed. Even for the hard puzzles, we only need to infer 1.6 times for each puzzle. This keeps the practical computational complexity of the algorithm in almost polynomial time.

	Solved	Unsolved	IT*	AI*
r-Reduction only	7050(70.5%) GC* 1.75226	2950(29.5%) GC* 1.75349	0	0
r-Reduction + Inference	10000(100%)	0	16034	1.6034

* IT: Inference Times. AI: Average Inferences. GC: Game Complexity.

Table 2. Results of hard puzzles

It is believed that all the Sudoku puzzles can be solved with only heuristic reasoning, without backtracking inferences. However, if only using r-Reduction, 70.5% of the hard puzzles are solved. So, for the future works, we are trying to find a stronger rule by extending r-Reduction, in order to solve all the puzzles using a single rule.

In fact, we developed an approach based on heuristic reasoning on certain discrete structure (graph), to solve a constraint satisfiability problem. This approach may provide a new method and concrete example for solving constraint-based problems. And it could also lead to new techniques for developing tutoring systems for Sudoku learners [1].

We also proposed game complexity to quantitatively measure the difficulty levels of Sudoku puzzles. The concept can be interpreted very well in the players' viewpoints, and it is also shown to be effective in experiments. It could also provide a potential measurement of uncertainty in reasoning.

References

[1] A. Caine and R. Cohen, "MITS: A Mixed-Initiative Intelligent Tutoring System for Sudoku," in *Luc Lamontagne, Mario Marchand (Eds.), Advances in Artificial Intelligence, Proceedings of 19th Conference of the Canadian Society for Computational Studies of Intelligence, Canadian AI 2006, LNCS vol. 4013*, 2006, pp. 550–561.

[2] B. Felgenhauer, F. Jarvis, "Enumerating possible Sudoku grids," 2005.
http://www.shef.ac.uk∼pm1afj/sudoku/sudoku.pdf.

[3] B. Felgenhauer, F. Jarvis, "Mathematics of Sudoku I," 2006.
http://www.sheffield.ac.uk∼pm1afj/maths/felgenhauer_jarvis_spec1.pdf.

[4] Krazydad, "Free Sudokus by Krazydad".
http://www.krazydad.com/sudoku/

[5] M. Henz and H.M. Truong, "SUDOKUSAT-A Tool for Analyzing Difficult Sudoku Puzzlesm," in *proceedings of the First International Workshop on Applications with Artificial Intelligence, IWAAI 2007*, 2007.

[6] A. Inkala, *AI Escargot - The Most Difficult Sudoku Puzzle*, Lulu.com Publisher, Finland, 2007.

[7] I. Lynce and J. Ouaknine, "Sudoku as a SAT problem," in *Electronic Proceedings of the 9th International Symposium on Artificial Intelligence and Mathematics*, 2006.

[8] G. Royle, *Minimum Sudoku*. http://people.csse.uwa.edu.au/gordon/sudokumin.php

[9] Ravel, *The hardest Sudokus*. http://www.sudoku.com/forums/viewtopic.php?t=4212&start=587

[10] C.G. Reeson, K.-C. Huang, K.M. Bayer, and B.Y. Choueiry, "An Interactive Constraint-Based Approach to Sudoku," in *proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (AAAI'07)*, 2007, pp. 1976–1977.

[11] E. Russell and F. Jarvis, "Mathematics of Sudoku II," 2006. http://www.afjarvis.staff.shef.ac.uk/sudoku/russell_jarvis_spec2.pdf

[12] C.E. Shannon, "A Mathematical Theory of Communication," *The Bell System Technical Journal, 27*, pp. 379–423, 623–656, 1948.

[13] H. Simonis, "Sudoku as a constraint problem," in *Hnich, B., Prosser, P., Smith, B. (eds.) Proceedings of the 4th International Workshop on Modelling and Reformulating Constraint Satisfaction Problems*, 2005, pp. 13–27.

[14] T. Weber, "A SAT-based Sudoku solver," in *Geooff Sutcliffe and Andrei Voronkov (eds.), proceedings of the 12th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR-12)*, 2005, pp. 11–15.

[15] T. Yato, and T. Seta, "Complexity and completeness of finding another solution and its application to puzzles," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science 86-A(5)*, pp. 1052–1060, Oxford University Press, 2003.